SEMANTICS
and
EARTH SCIENCE MARKUP LANGUAGE

"Define Once Use Anywhere"

RAHUL RAMACHANDRAN, SUNIL MOVVA
HELEN CONOVER and SARA GRAVES

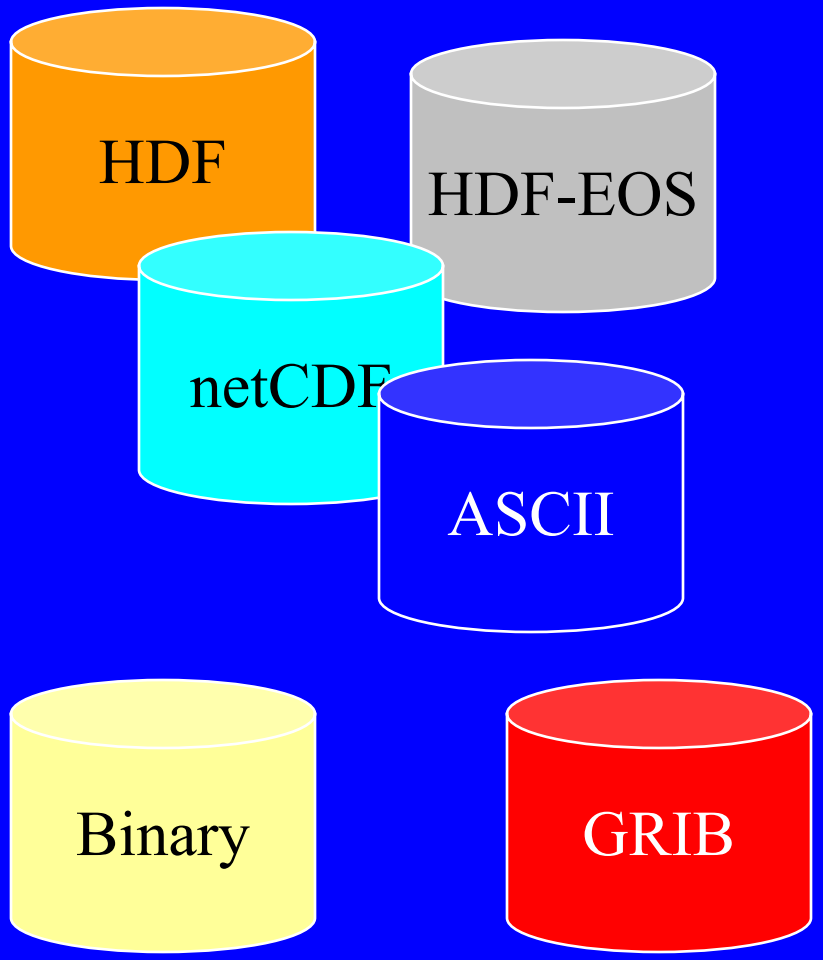
INFORMATION TECHNOLOGY AND SYSTEMS CENTER
UNIVERSITY OF ALABAMA IN HUNTSVILLE

Presentation Overview

- ESML Vision
  - Data/Application Interoperability Problem
  - Interchange Technology Solution via ESML
- ESML v3.0 Details
  - Changes to Schema, Library
  - Schema, Library Description
  - Other new additions
- Semantics and ESML
  - Embedding semantics in ESML files using ontologies
  - Example: Smart Subsetter Prototype

ESML Vision

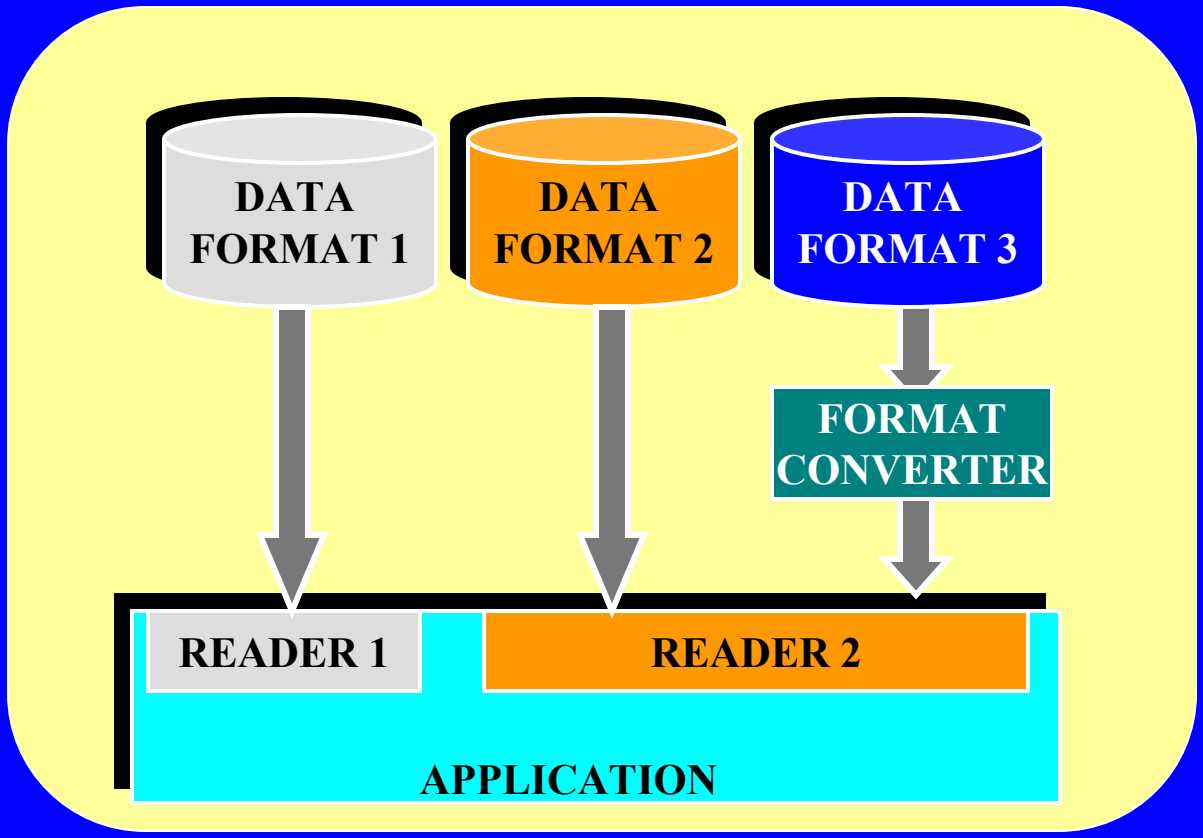
Earth Science Data Characteristics



- Different formats, types and structures (18 and counting for Atmospheric Science alone!)
- Different states of processing (raw, calibrated, derived, modeled or interpreted)
- Enormous volumes

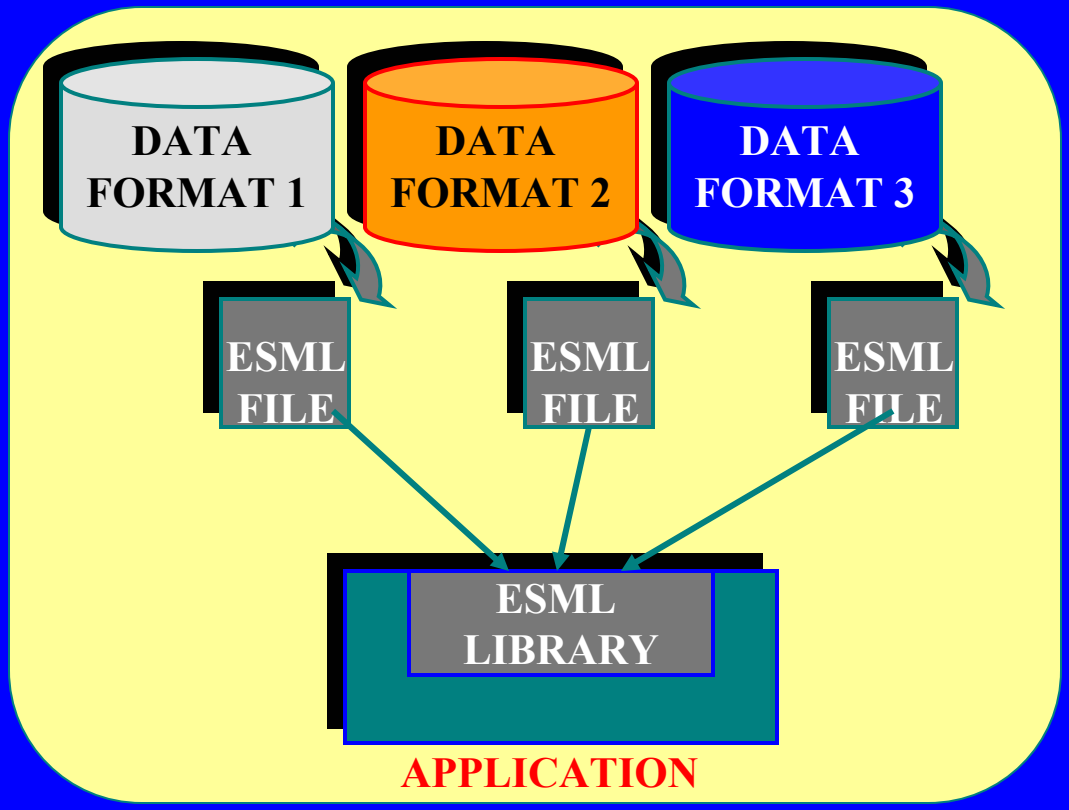
- **Heterogeneity leads to Data usability problem**

Data Usability Problem



- Requires specialized code for every format
  - Difficult to assimilate new data types
  - Makes applications tightly coupled to data
- One possible solution - enforce a Standard Data Format
  - Not practical for legacy datasets

ESML Solution

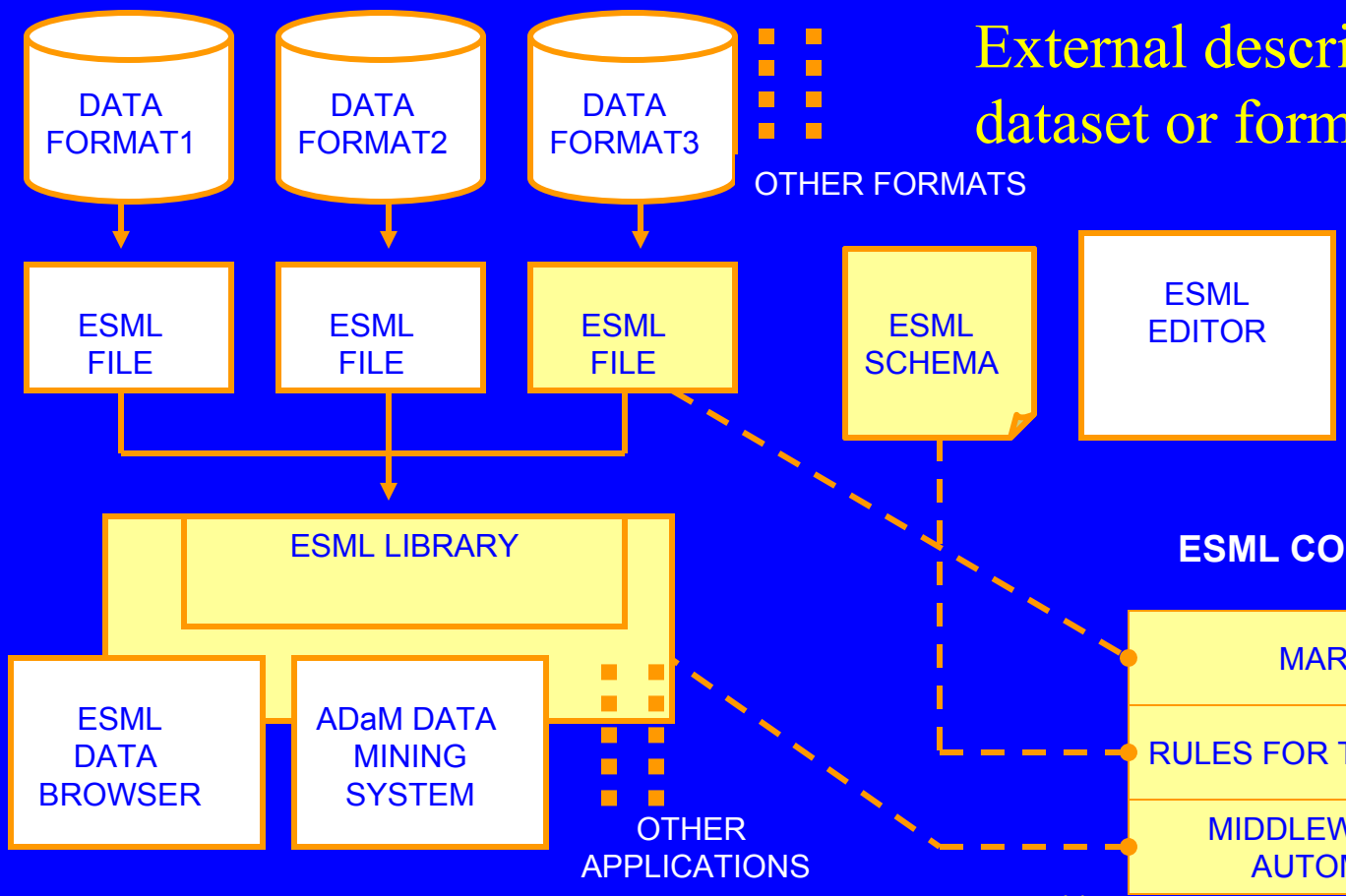


- ESML (external metadata) files containing the structural description of the data format
- Applications utilize these descriptions to figure out how to read the data files resulting in data interoperability for applications

What is ESML?

- It is a specialized markup language for Earth Science metadata based on XML
- It is a machine-readable and -interpretable representation of the structure and content of any data file, regardless of data format
- ESML description files contain external metadata that can be generated by either data producer or data consumer (at collection, data set, and/or granule level)
- ESML provides the benefits of a standard, self-describing data format (like HDF, HDF-EOS, netCDF, geoTIFF, …) without the cost of data conversion
- ESML is the basis for core Interchange Technology that allows data/application interoperability

Components of the ESML Interchange Technology

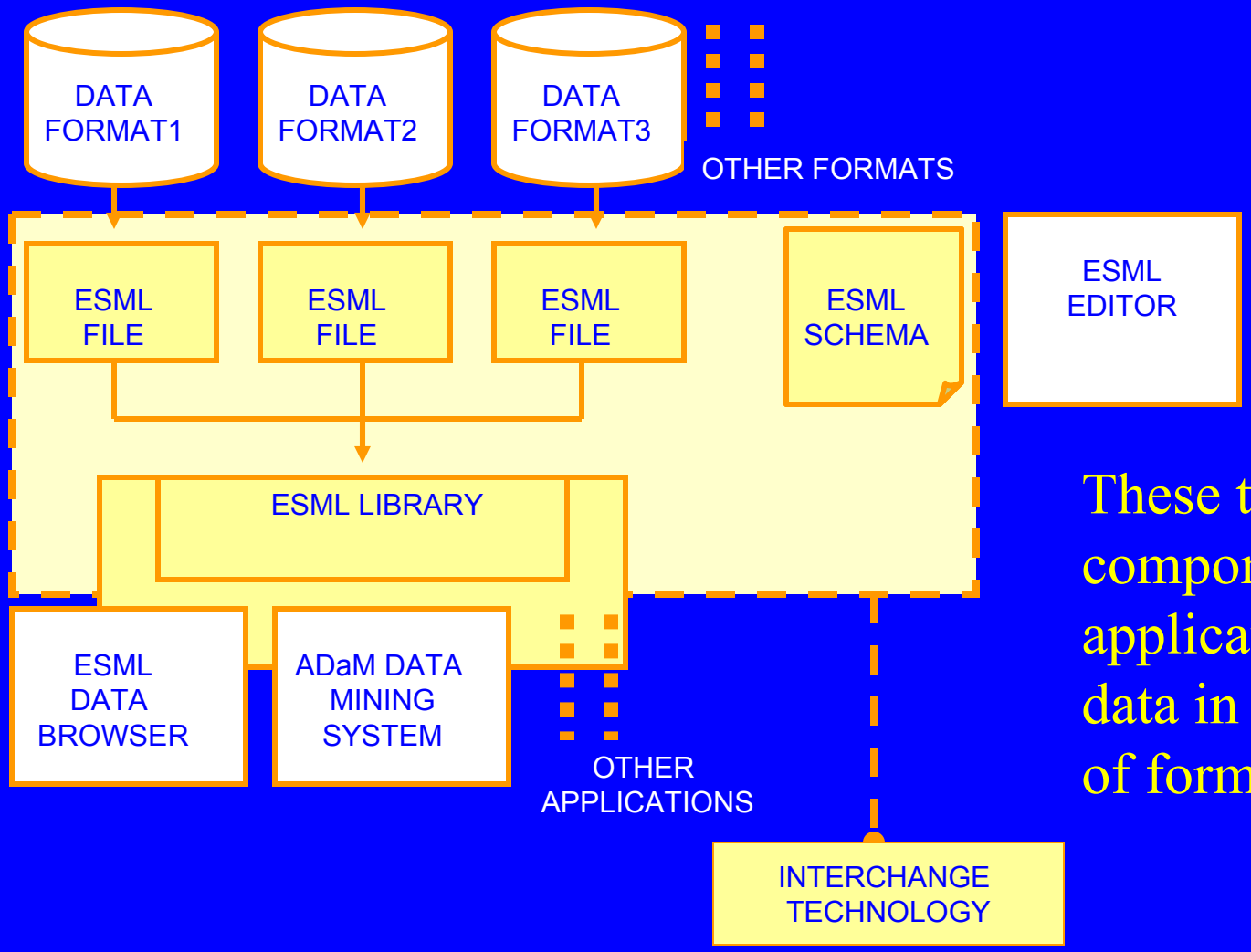


External description file for dataset or formats

Rules that govern the description of the data files

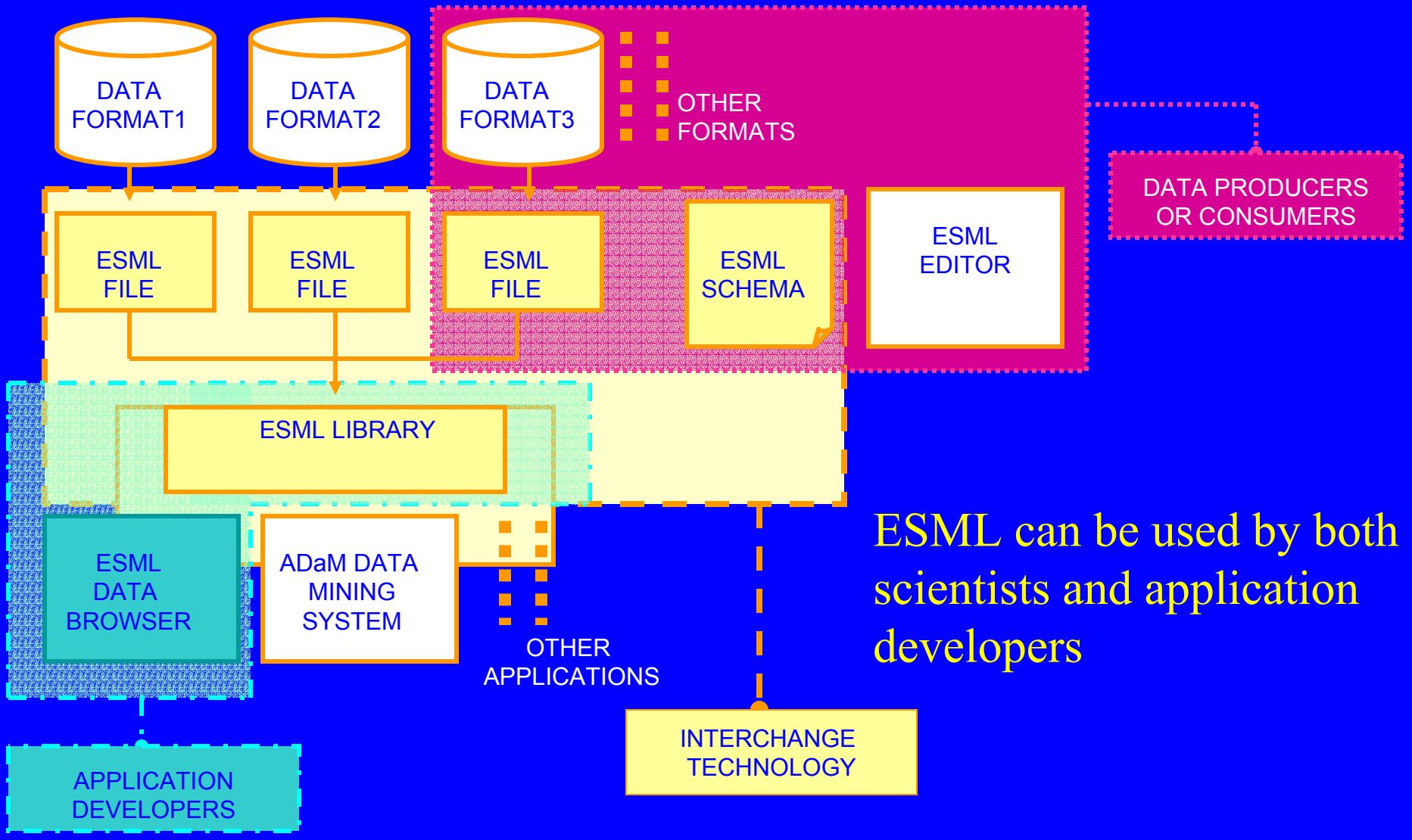
Library parses and interprets the description file and figures out how to read the data

Components of the ESML Interchange Technology



These three key components allow applications to use data in a wide variety of formats

Interchange Technology for Data Users and Application Developers



ESML can be used by both scientists and application developers

Advantages of using ESML

- Scientist (Data Producer/Consumer)
  - ESML will let them use virtually any data format in their applications
  - ESML files are external description files that can be easily created, modified and viewed by any text editor
  - ESML has a few simple concepts which can be used to describe numerous data sets
  - An ESML file can be seen as a set of instructions to the application on how to read and understand a data file
  - If the format of the data changes for whatever reason (e.g., new version of data set) no software changes are required, just a new ESML file.
- Does that mean a scientist has to write an ESML file for every data file?
  - No, in fact the beauty of ESML is that it allows scientist to write ONE ESML file to describe MANY data files that are structurally and semantically similar

Advantages of using ESML

- Data Archiving Centers (Data Producers)
  - ESML files can be used to store not only the *structural* but also embed *semantic* information about the data sets
  - Since ESML files are independent separate files, they can be generated on the fly utilizing metadata databases as datasets are ordered
  - Centers can archive data in its native formats and not have to store them in any "selected" format
  - Centers can now also "ESMLize" all their legacy datasets with minimal efforts
  - The existing legacy datasets now become a more valuable data resource for scientists, because they can be used more efficiently and effectively
- Application Developers
  - By using the ESML library, developers can build "ESML enabled" applications!
  - ONE single reader component can read all the various data formats instead of having separate reader module for different formats

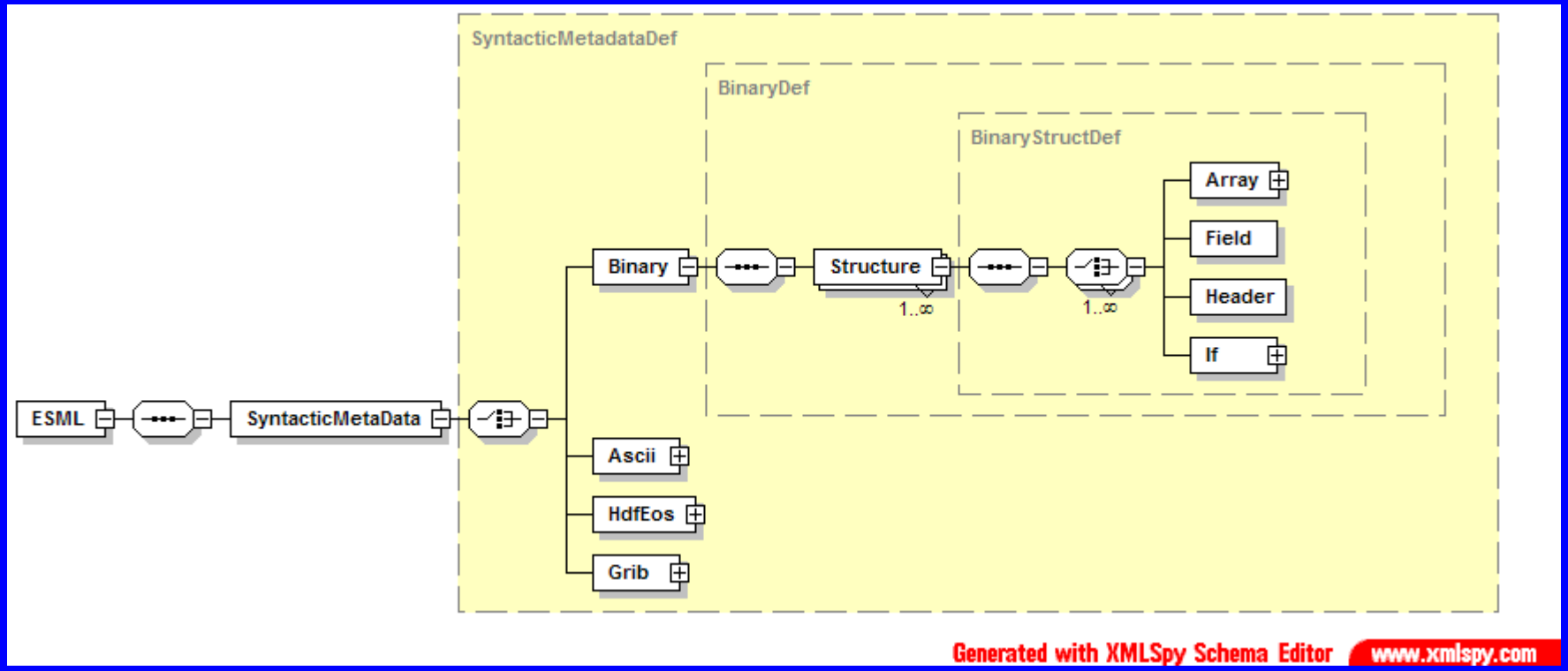
ESML v3.0

Changes ESML v3.0 Schema

- Removed the embedded semantics such as the Latitude, Longitude, Data tags, etc., from the schema
- Contains a new tag <Header> which is identical to <Field> but semantically used for comments and header information including symbols
- Allows external semantics to be embedded within the structural definitions

ESML v3.0 Schema

- ESML schema defines ***Syntactic*** metadata that describe the structure of the file in machine-readable and -interpretable terms



Writing an ESML File (1)

ESML MARKUP
FOR THE DATA FILE

```
<a:ESML >
 <SyntacticMetaData>
  <Ascii>
   <Structure instances="1">
    <Header name="SizeX" format="%d" />
    <Header name="SizeY" format="%d" />
    <Array occurs="4">
     <Array occurs="5">
      <Field name="BrightnessTemp" format="%d"/>
     </Array>
    </Array>
   </Structure>
  </Ascii>
 </SyntacticMetaData>
</a:ESML>
```

The next slides will set describe how to write an ESML file for a simple ASCII file described below

```
 4
 5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
```

SIMPLE ASCII
DATA FILE

Writing an ESML File (2)

DESCRIBING ONLY THE STRUCTURE

```
<a:ESML >
  <SyntacticMetaData>
    <Ascii>
      <Structure instances="1">
        <Header name="SizeX" format="%d" />
        <Header name="SizeY" format="%d" />
        <Array occurs="4">
          <Array occurs="5">
            <Field name="BrightnessTemp" format="%d"/>
          </Array>
        </Array>
      </Structure>
    </Ascii>
  </SyntacticMetaData>
</a:ESML>
```

```
 4
 5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
```

SIMPLE ASCII DATA FILE

Writing an ESML File (3)

DESCRIBE THE FORMAT

```
<a:ESML >
  <SyntacticMetaData>
    <Ascii>
      <Structure instances="1">
        <Header name="SizeX" format="%d" />
        <Header name="SizeY" format="%d" />
        <Array occurs="4">
          <Array occurs="5">
            <Field name="BrightnessTemp" format="%d"/>
          </Array>
        </Array>
      </Structure>
    </Ascii>
  </SyntacticMetaData>
</a:ESML>
```

```
 4
 5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
```

SIMPLE ASCII DATA FILE

Writing an ESML File (4)

ENTIRE FILE CONTENTS INTO 1 LOGICAL STRUCTURE

```
<a:ESML >
  <SyntacticMetaData>
    <Ascii>
      <Structure instances="1">
        <Header name="SizeX" format="%d" />
        <Header name="SizeY" format="%d" />
        <Array occurs="4">
          <Array occurs="5">
            <Field name="BrightnessTemp" format="%d"/>
          </Array>
        </Array>
      </Structure>
    </Ascii>
  </SyntacticMetaData>
</a:ESML>
```

```
4
5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
```

SIMPLE ASCII DATA FILE

Writing an ESML File (5)

DEFINE THE FIRST FIELD IN THE FILE: HEADER INFORMATION

```
<a:ESML >
  <SyntacticMetaData>
    <Ascii>
      <Structure instances="1">
        <Header name="SizeX" format="%d" />
        <Header name="SizeY" format="%d" />
        <Array occurs="4">
          <Array occurs="5">
            <Field name="BrightnessTemp" format="%d"/>
          </Array>
        </Array>
      </Structure>
    </Ascii>
  </SyntacticMetaData>
</a:ESML>
```

```
4
5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
```

SIMPLE ASCII DATA FILE

Writing an ESML File (6)

DEFINE THE SECOND FIELD IN THE FILE: HEADER INFORMATION

```
<a:ESML >
  <SyntacticMetaData>
    <Ascii>
      <Structure instances="1">
        <Header name="SizeX" format="%d" />
        <Header name="SizeY" format="%d" />
        <Array occurs="4">
          <Array occurs="5">
            <Field name="BrightnessTemp" format="%d"/>
          </Array>
        </Array>
      </Structure>
    </Ascii>
  </SyntacticMetaData>
</a:ESML>
```

```
4
5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
```

SIMPLE ASCII DATA FILE

Writing an ESML File (7)

DEFINE THE DATA FIELD IN THE FILE: PROVIDE SIZE AND FORMAT INFORMATION

```
<a:ESML >
 <SyntacticMetaData>
  <Ascii>
   <Structure instances="1">
    <Header name="SizeX" format="%d" />
    <Header name="SizeY" format="%d" />
    <Array occurs="4">
     <Array occurs="5">
      <Field name="BrightnessTemp" format="%d"/>
     </Array>
    </Array>
   </Structure>
  </Ascii>
 </SyntacticMetaData>
</a:ESML>
```

```
4
5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
```

SIMPLE ASCII DATA FILE

Writing an ESML File (8)

CLOSE ALL THE TAGS: ESML FILE IS READY

```
<a:ESML >
 <SyntacticMetaData>
  <Ascii>
   <Structure instances="1">
    <Header name="SizeX" format="%d" />
    <Header name="SizeY" format="%d" />
    <Array occurs="4">
     <Array occurs="5">
      <Field name="BrightnessTemp" format="%d"/>
     </Array>
    </Array>
   </Structure>
  </Ascii>
 </SyntacticMetaData>
</a:ESML>
```

```
 4
 5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
```

SIMPLE ASCII DATA FILE

Another Possible ESML Description (9)

USE HEADER INFORMATION

```
<a:ESML >
  <SyntacticMetaData>
    <Ascii>
      <Structure instances="1">
        <Header name="SizeX" format="%d" symbol="true" />
        <Header name="SizeY" format="%d" symbol="true" />
        <Array occurs="$SizeX">
          <Array occurs="$SizeY">
            <Field name="BrightnessTemp" format="%d"/>
          </Array>
        </Array>
      </Structure>
    </Ascii>
  </SyntacticMetaData>
</a:ESML>
```

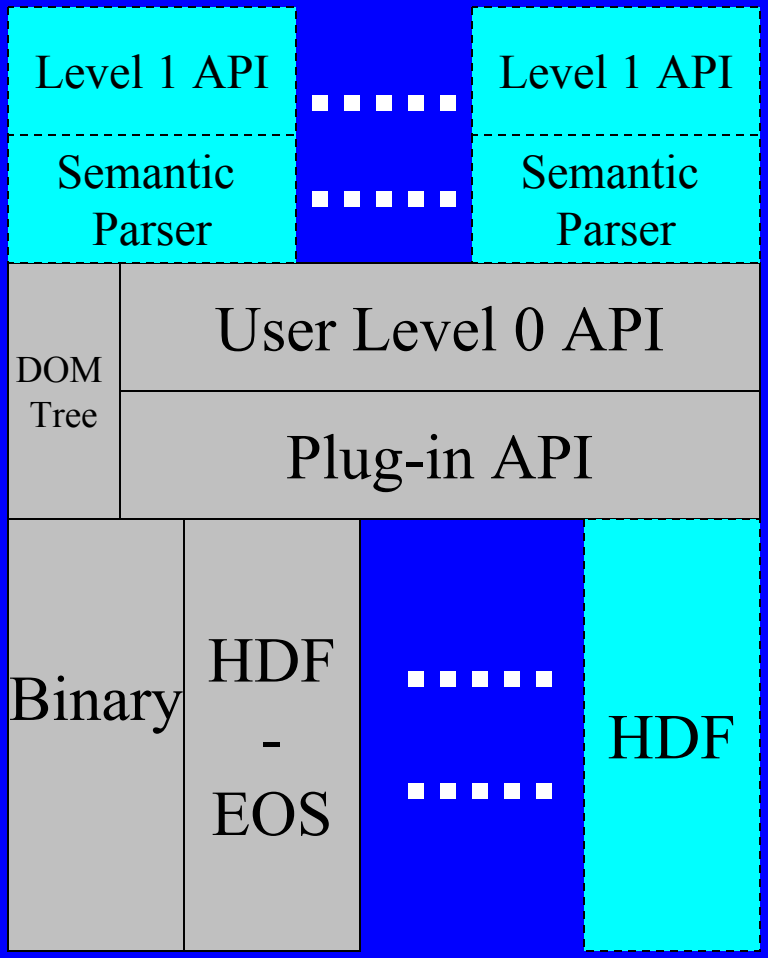
```
 4
 5
 1  2  3  4  5
 6  7  8  9 10
11 12 13 14 15
16 17 18 19 20
```

SIMPLE ASCII DATA FILE

Changes in ESML v3.0 Library

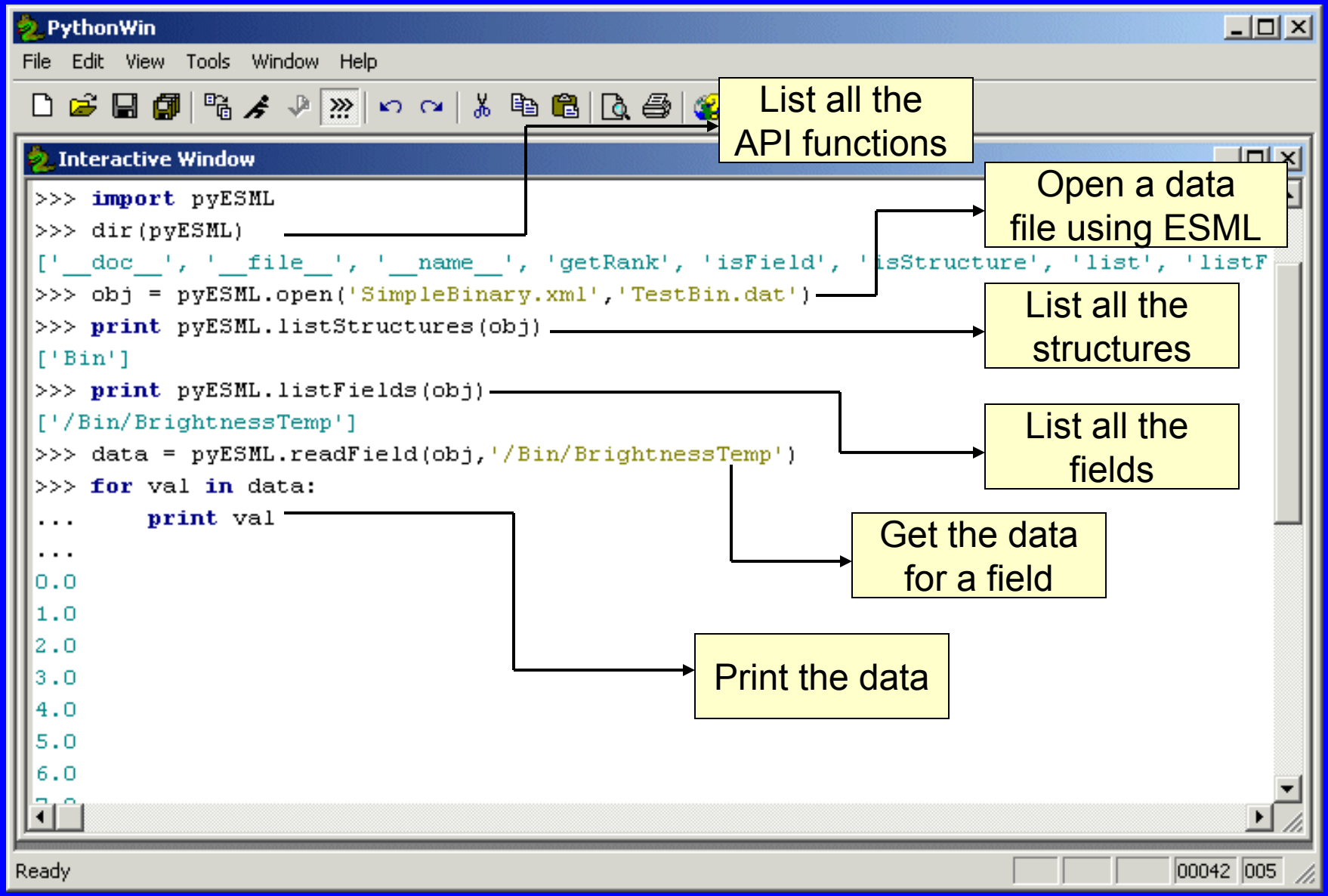
- Design for the refactored library follows the layered cake approach where:
  - Lowest (core) level provides the basic functionality of reading the structural metadata from the ESML file and returning data to the user
  - Additional software layers can be added to provide other functionalities such as using semantics from an ontology to "use" the data intelligently
- Includes plug-in modules for each individual format, allowing packaging of libraries
- Provides a simple API for easy addition of new formats as plug-in modules
- Provides a more intuitive user API based on the analogy of file access in a directory structure
- Provides the library source code via Source Forge repository

ESML v3.0 Library Design

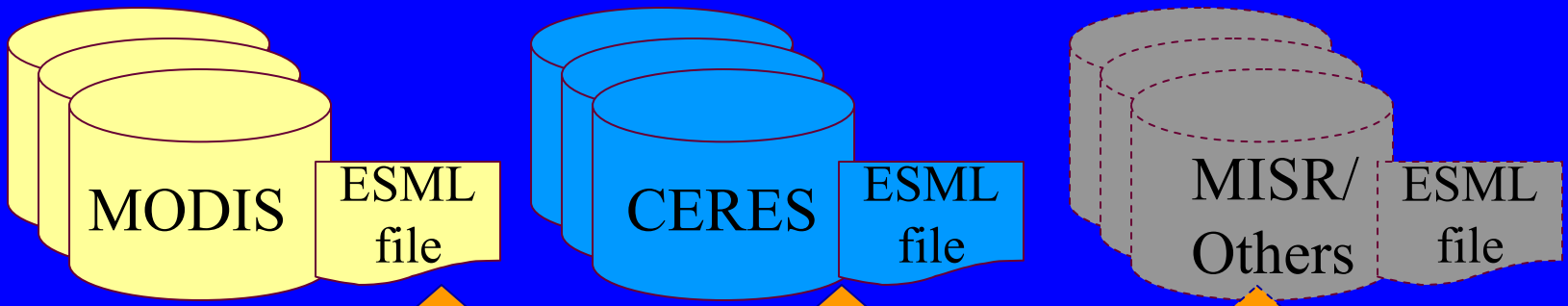


- Layered Cake Design
- Additional modules for other formats can be easily added
- Additional layers of functionality can be easily added on top of the Core Library (shown in grey)
- Intuitive user API based on the analogy of file access in a directory structure
- Versions Available:
  - C++ for Windows and Linux
  - Python (pyESML)

PyESML Example



ESML IN ACTION: Collocation Algorithm



Purpose:

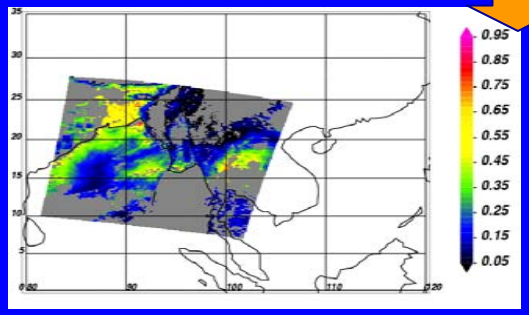
- *To study the relationship between shortwave flux and cloud/aerosol properties*
- *Important for climate change studies*

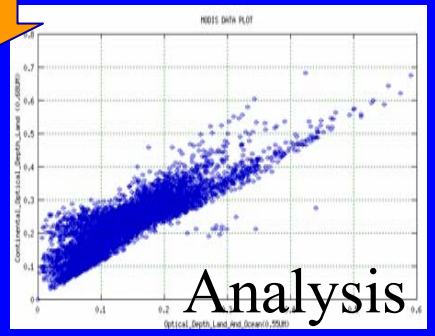
ESML Library

Collocation Algorithm

Scientists can:

- *Select a variety of data in different formats for the collocation analysis*





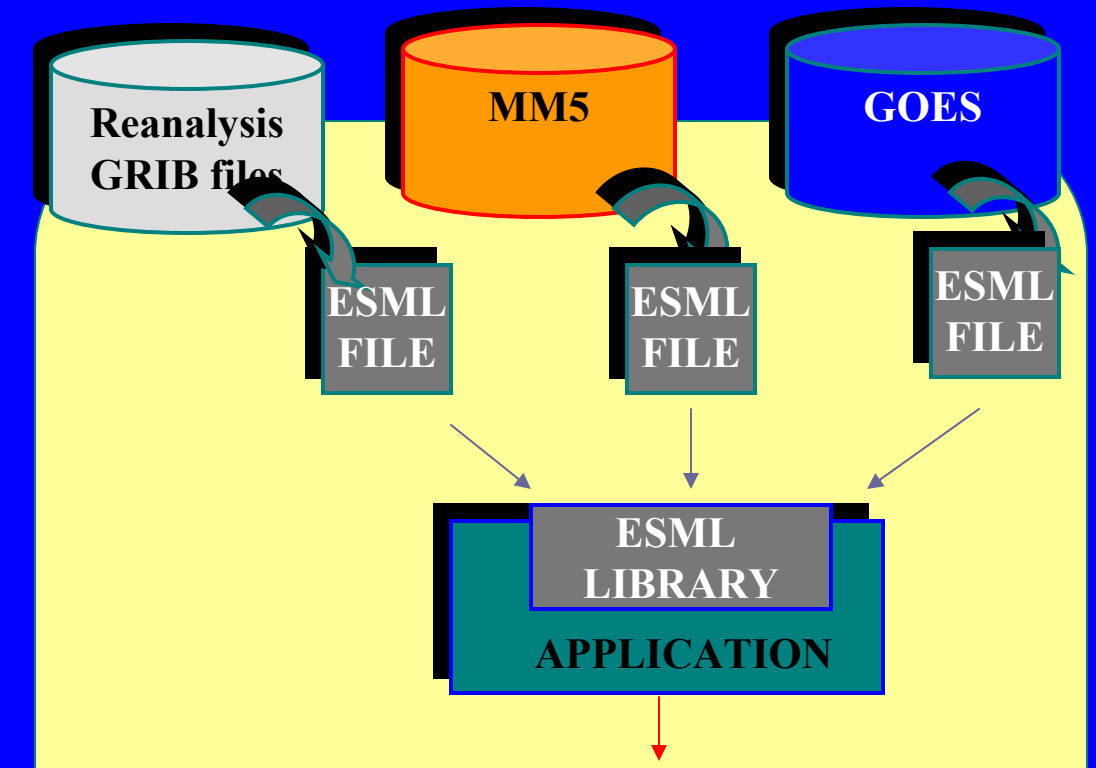
Analysis

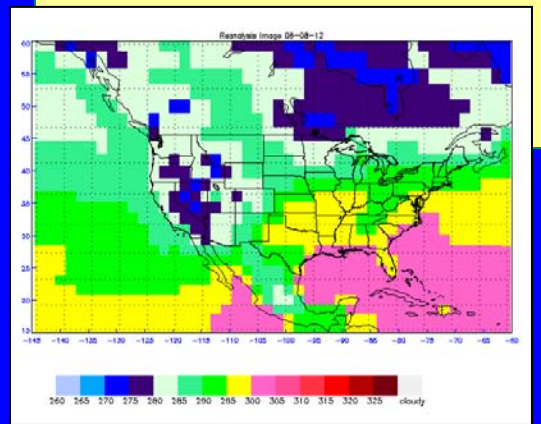
ESML IN ACTION:

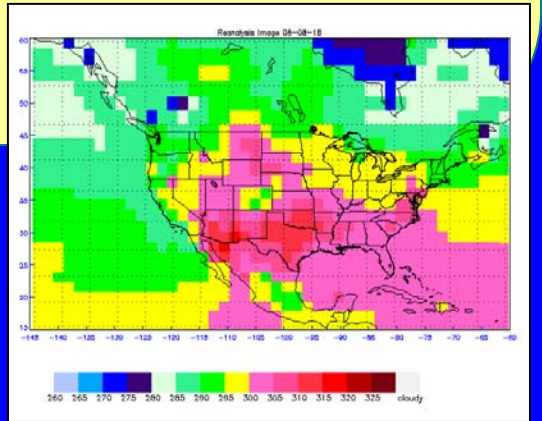
Ingest surface skin temperature data in Numerical Models

Skin temperatures come in a variety of data formats -

- •GOES - McIDAS
- •Reanalysis Data - GRIB
- •MM5 Model - MM5 Binary
- •AVHRR - HDF
- •MODIS - EOS-HDF







Semantics and ESML

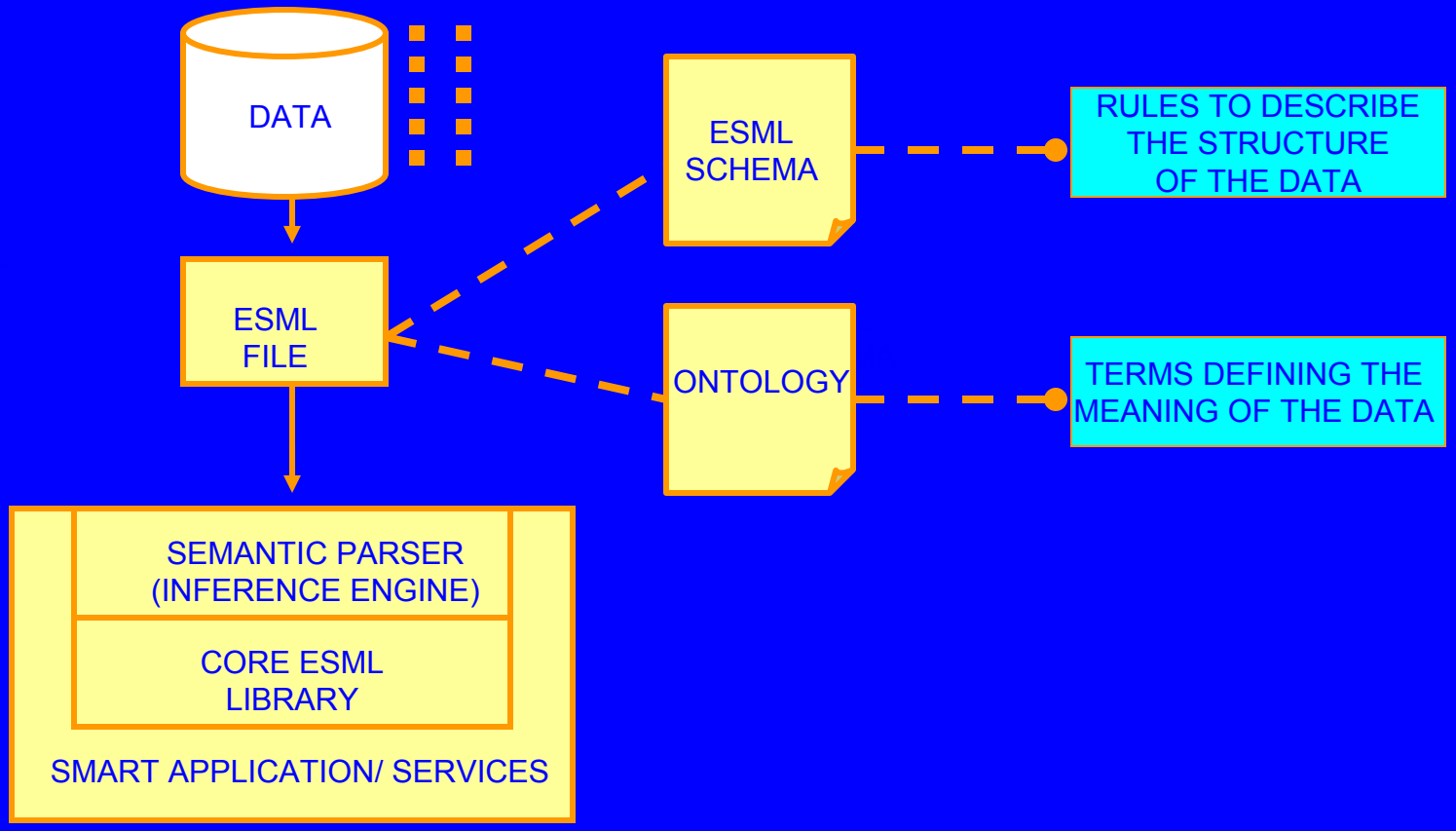
Ontology

- Many different definitions depending upon view points and application field
- Most commonly used one in Machine Learning/AI/Intelligent Systems: "An Ontology is a FORMAL, EXPLICIT specification of a SHARED conceptualization" [Gruber, 1993]
  - Explicit – type of concept and constraints of use are explicitly defined
  - Formal – should be machine understandable
  - Shared – captures consensual knowledge
- Ontology consists of concepts and their relationships
    - All Men are Mortal – is a concept!
    - Socrates is a Man – is not a concept but a instance hence will not be in an ontology
- Relationships can be defined for different concepts
  - Tall is a "subclass" of Man

Semantics in ESML

- ESML schema's focus is on providing structural data interoperability between data/application
- ESML will allow embedding semantic terms for data fields in the description file to provide complete structural and semantic description of the data
- ESML will allow linking to appropriate ontologies
- Various science communities can create their own ontologies and link them with ESML description files for their data
- Application developers can add semantic parsers on top of the core ESML library to build ***"smart" applications/services***

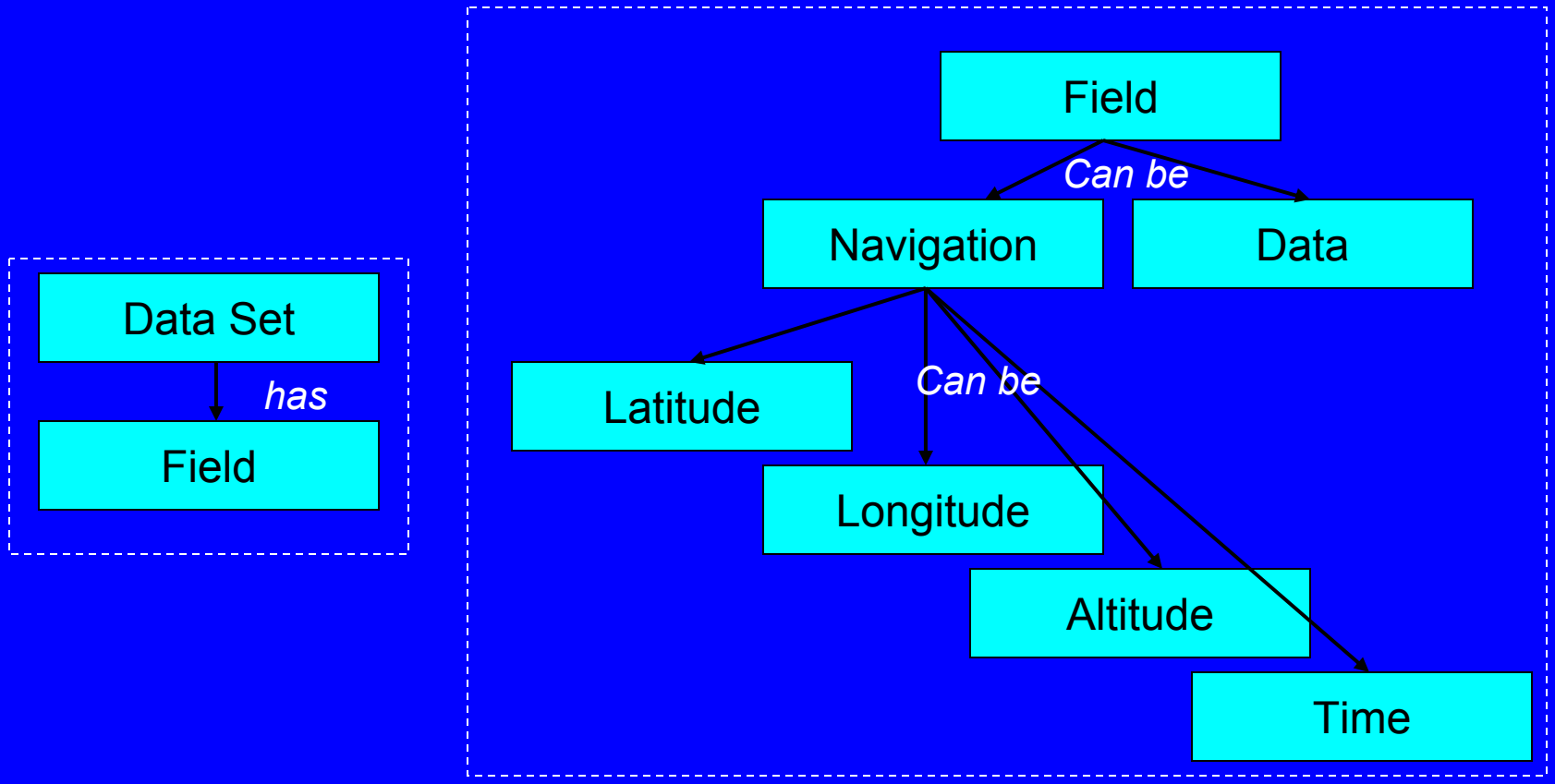
Smart Applications/Services using ESML Schema and Ontology



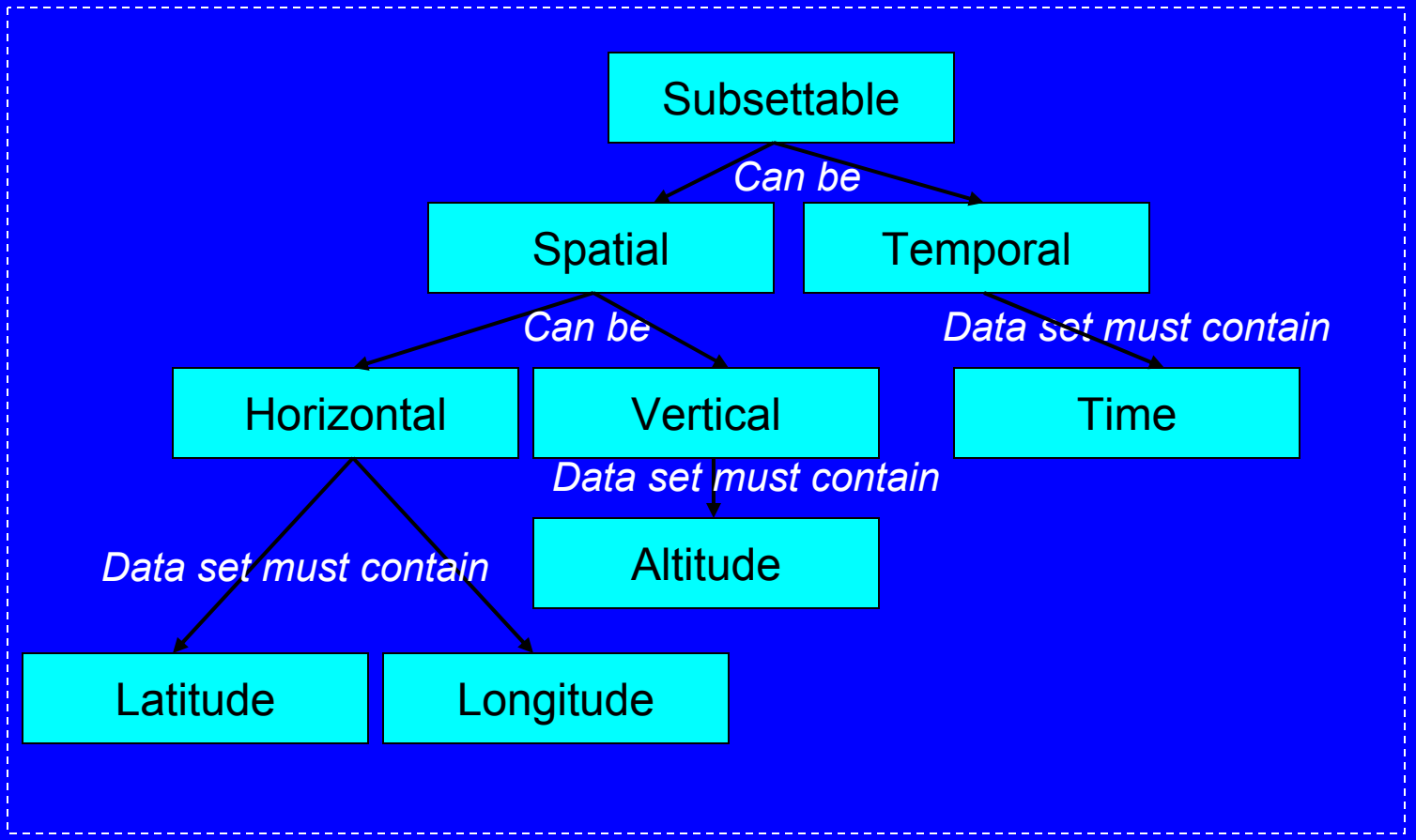
Proof of Concept Prototype for a Smart Subsetter

- Prototype will:
  - Parse the semantic tags embedded in the ESML file
  - Use the linked ontology to decipher meaning of these tags
  - Make useful decisions
- Components of the Prototype
  - Simple ontology describing “Subsetting”
  - ESML description files
  - Reasoning System from JTP from Stanford University used as an inference engine

Dataset Ontology



Subsetting Ontology



ESML Example Files

Example 1: SampleSet

```
<Latitude rdf:ID="Lat"/>
<Longitude rdf:ID="Lon"/>
<DataField rdf:ID="BrightnessTemp"/>

<DataSet rdf:ID="SampleSet">
        <hasField rdf:resource="#Lat"/>
        <hasField rdf:resource="#Lon"/>
        <hasField rdf:resource="#BrightnessTemp"/>
</DataSet>
```

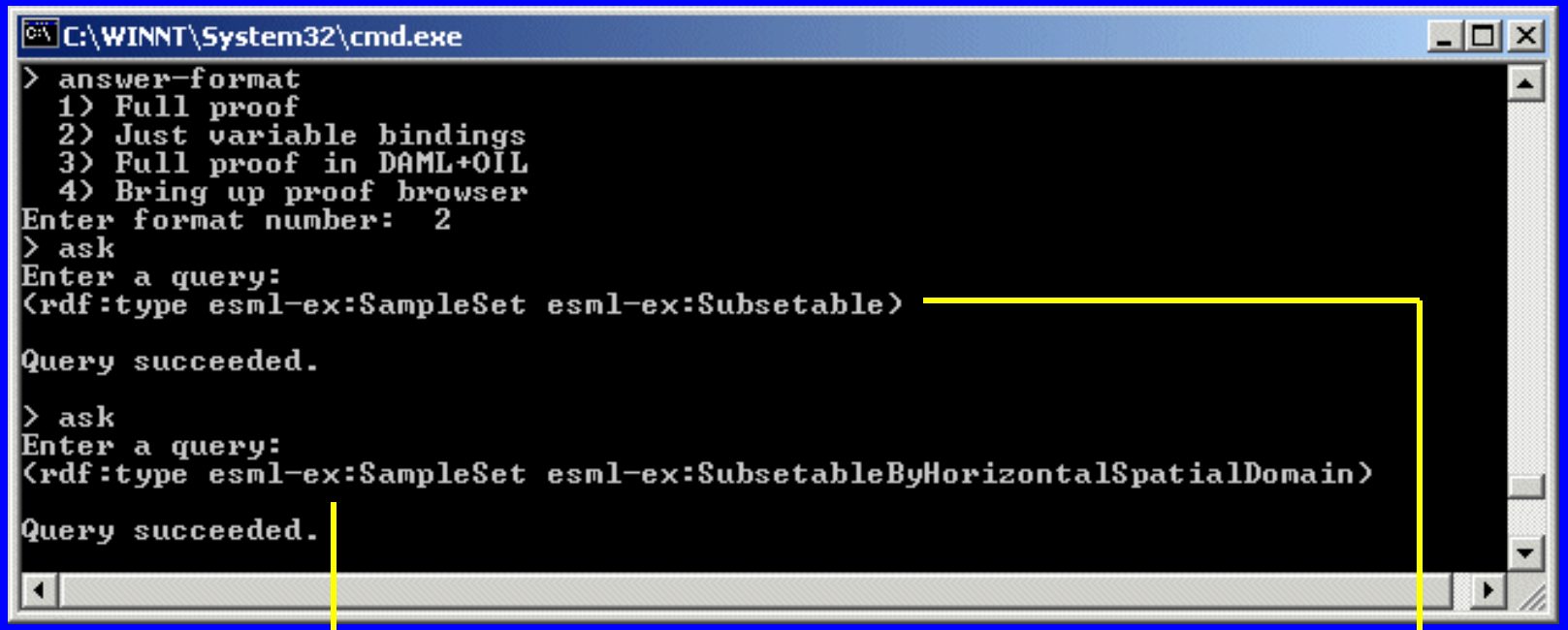
Data set contains instances of: Latitude, Longitude, Data

Example 2: SampleSet2

```
<Latitude rdf:ID="Lat1"/>
<Longitude rdf:ID="Lon1"/>
<Time rdf:ID="time1"/>
<DataField rdf:ID="WindSpeed"/>
<DataSet rdf:ID="SampleSet2">
        <hasField rdf:resource="#Lat1"/>
        <hasField rdf:resource="#Lon1"/>
        <hasField rdf:resource="#WindSpeed"/>
        <hasField rdf:resource="#time1"/>
</DataSet>
```

Data set contains instances of: Latitude, Longitude, Time And Data

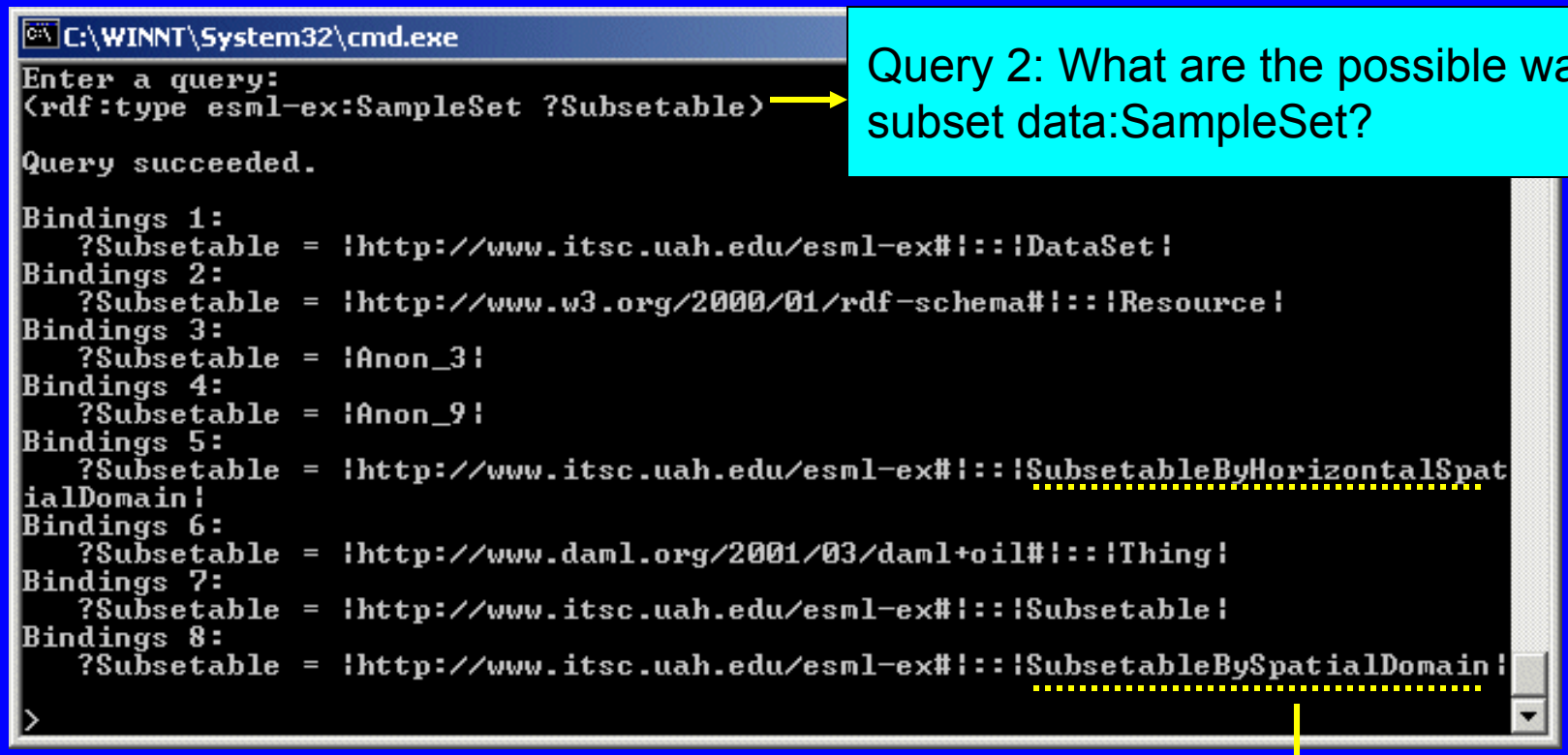
Querying the Inference Engine



Query 1: Is the data:SampleSet subsettable?
Answer: Yes

Query 2: Is the data:SampleSet subsettable spatially ?
Answer: Yes

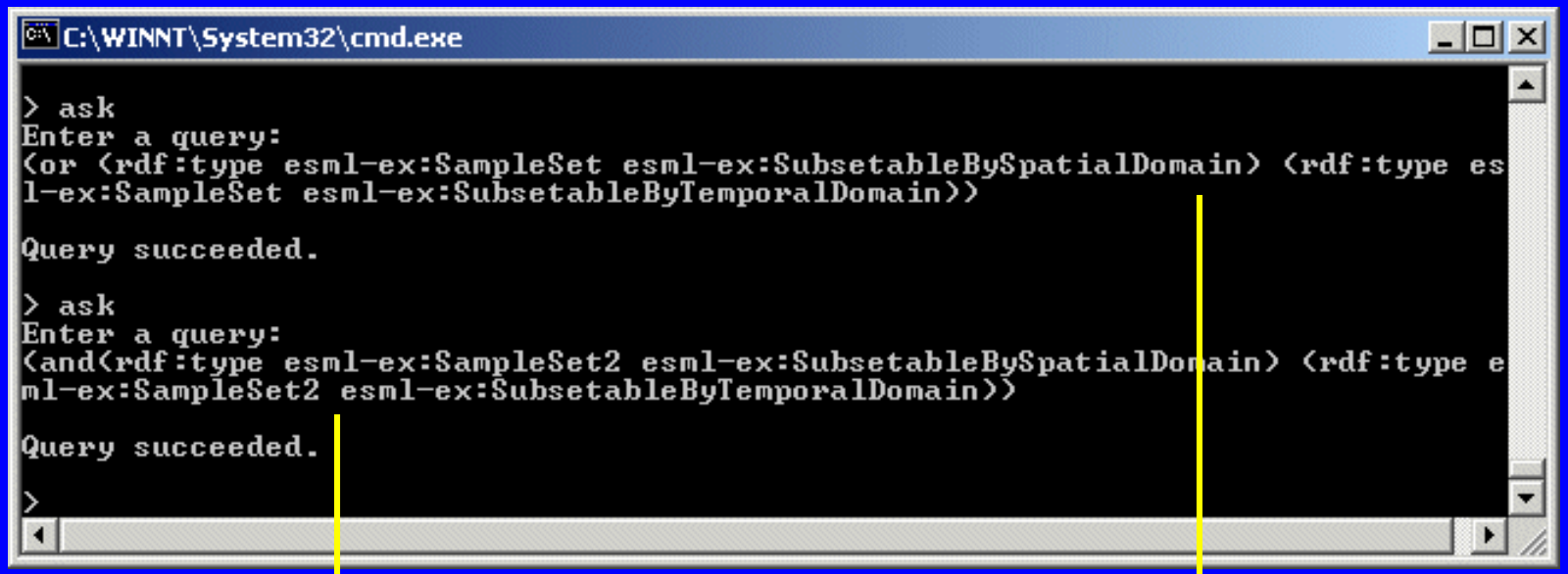
Querying the Inference Engine



Query 2: What are the possible ways to subset data:SampleSet?

Answer: Spatially based on horizontal navigation fields

Querying the Inference Engine



Query 1: Is the data:SampleSet subsettable either spatially or temporally?
Answer: Yes

Query 2: Is the data:SampleSet2 subsettable spatially and temporally ?
Answer: Yes

Summary

- ESML is not a new data format
- ESML enables independently developed applications and services to effectively utilize wide variety of heterogeneous data products
- Whats new in ESML v3.0?
  - Changes to ESML schema and library design
  - LINUX, Windows versions and the source code available
  - Python wrappers for the library
- New design will enable embedding semantics defined in external domain ontologies in ESML files
- ***Combing ESML and Ontologies will allow the development of "smart" applications/services/tools***

For More Information

- URL: **esml.itsc.uah.edu**
- Become a member and post ESML related news items on the website
- Schema and related documents available to all
- Download the latest products
- Source code available via Source Forge
- Join the ESML mailing list